1. Sta ce biti prikazano na standardnom izlazu nakon izvrsavanja sledeceg koda:  
  
 vector<string> spisak;

spisak.push\_back("Beba");

spisak.push\_back("Marko");

spisak.push\_back("Daca");

spisak.push\_back("Milan");

for (unsigned i = 0; i < spisak.size(); i++)

{

cout << spisak[i]<< " - ";

}

* **Beba - Marko - Daca - Milan -**

2.Koji je rezultat izvrsavanja sledeceg programa:

int vred = 30;

try

{

if (vred < 20) throw vred;

}

catch (int vred)

{

cout << "U klauzuli catch." << endl;

}

ue << endl;

return 0;

* **Program ima gresku prevodjenja.**

3. Sta ce biti prikazano na standardnom izlazu nakon izvrsavanja sledeceg koda:

vector<int> ListaCelih(5);

int n = 1;

for (auto &p: ListaCelih)

{

if (n % 2 == n % 3)

p = 2 \* n;

else

p = 3 \* n;

n = p - 2;

}

for (auto p: ListaCelih)

{

cout << p << " ";

}

cout << endl<<endl;

* **2 0 -6 -24 -78**

4. Sta ce biti prikazano na standardnom izlazu nakon izvrsavanja sledeceg koda:

vector<int> intList(7);

intList[0] = 5;

for (int i = 1; i < 7; i++)

intList[i] = 2 \* intList[i - 1] + i;

for (int i = 0; i < 7; i++)

cout << intList.at(i) << " ";

* **5 11 24 51 106 217 440**

5.Oznacite linije koda koje su pogresne:

#include <iostream>

using namespace std;

int main()

{

int n1{ 300 };

int n2{ 10 };

try

{

**kol = n1 / n2;**

}

**cout << "Kolicnik je " << kol << endl;**

catch (string exceptionString)

{

cout << exceptionString;

}

return 0;

}

**#linije koje su pogresne su oznacene ovako**

5.Oznacite linije koda koje su pogresne:

#include <iostream>

using namespace std;

template<class T>

T square(T number)

{

**return T \* T;**

}

int main()

{

double n{ 3.14 };

cout << square(n) << endl;

return 0;

}

6.Sta se ispisuje na izlazu nakon izvrsavanja sledeceg programa:

#include <iostream>

#include <vector>

using namespace std;

template<class Iterator>

void double\_each\_element(Iterator begin, Iterator end) {

for (auto it = begin; it != end; ++it) {

\*it \*= 2;

}

}

int main() {

std::vector<int> v{ 1, 2, 3, 4, 5, 6 };

double\_each\_element(v.begin(), v.end()); // 1. Prolaz: ceo vektor

double\_each\_element(v.begin(), v.begin() + 3); // 2. Prolaz: prva tri elementa

double\_each\_element(&v[0], &v[3]); // 3. Prolaz: prva tri elementa (isto kao i 2. prolaz)

for (auto e : v) cout << e << " "; // Ispis vektora

cout << endl;

return 0;

}

* **8 16 24 8 10 12**

7.Oznacite linije koje sadrze greske u sledecem programskom kodu:

#include <iostream>

using namespace std;

void fun(double x) throw (double)

{

if (x < 10.0) throw 10.0;

}

int main() {

**fun(5); mozda ova ipak zato sto nema try catch**

return 0;

}

8.Sta se ispisuje na izlazu nakon izvrsavanja sledeceg programa:

template<class Container>

int count(const Container& container) {

int sum = 0;

for (auto&& e : container)

{

sum += 1;

}

return sum;

}

int main() {

std::vector<int> v1{ 3,1,4,1,5,9,2,6 };

std::vector<int> v2{ v1.begin()+1,v1.end()-2 };

int n1 = count(v1);

int n2 = count(v2);

cout << n1 << " " << n2;

}

* **8 5**

9. Osnovne kategorije kontejnera u jeziku C++ su

* **Asocijativni kontejneri**
* **Kontejnerski adapteri**
* **Kontejneri Sekvenci**

10. Osnovni operatori koji definisu ponasanje SVIH iteratora su:

* **--Pomera iterator na prethodni element**
* **== Proverava istu poziciju dva iteratora**
* **->Pristupa clanu elemenata na tekucoj poziciji**
* **\* Vraca element na tekucoj poziciji**
* **++ Pomera iterator na sledeci element**
* **= Dodaljuje vrednost iteratora**

11. Sta ce biti prikazano na standardnom izlazu nakon izvrsavanja sledeceg koda:

vector<int> ListaCelih(5);

for (int i = 0; i < 5; i++)

ListaCelih[i] = i \* (i + 1);

for (auto p : ListaCelih)

cout << p << " ";

* **0 2 6 12 20**

12. Standardna biblioteka sablona (STL)

* **Po obimu znatno manja od Standardne biblioteke jezika C++**
* **Predstavlja deo Standardne biblioteke jezika C++**

13.Osnovne komponente Standardne biblioteka sablona (STL) su

* **kontejneri**
* **iteratori**
* **algoritmi**
* **funkcijski objekti**

14.Regularni izrazi u jeziku C++

* **Postoje u STL biblioteci od verzije C++11.**
* **Postoje i mogu se koristiti preko zaglavlja <regex>.**

15.Sta ce biti prikazano na standardnom izlazu nakon izvrsavanja sledeceg koda:

vector<int> listaCelih(10);

for (int i = 0; i < 10; i++)

listaCelih[i] = 2 \* i + 5;

cout << listaCelih.front() << " " << listaCelih.back() << endl;

* **5 23**

16. Osnovne vrste (kategorije) iteratora su:

* **Jednosmerni(froward)**
* **Izlazni i ulazni**
* **Biditekcioni (bidirectional)**
* **S direktnim pristupom (random access)**

17.Sabloni STL kontejnerskih klasa definisani su sledecim zaglavljima

* **vector**
* **array**
* **deque**
* **list**
* **forward\_list**
* **map**
* **unordered\_map  
  set**
* **unordered\_set  
  bitset**

18.Koji je rezultat prevodjenja sledeceg programa?

#include <iostream>

using namespace std;

int main() {

int vred = 30;

try

{

if (vred > 30)throw vred;

}

cout << vred;

return 0;

}

* **Program ima gresku prevodjenja**

19.Rezultat izvrsavanja sledeceg programa (tacke u rezultatu oznacavaju prazno mesto - blank):

#include <iostream>

using namespace ::std;

int main() {

ios\_base::fmtflags original\_flags = cout.flags();

cout << 123 << "|";

cout.setf(ios\_base::left, ios\_base::adjustfield);

cout.width(5);

cout << 124 << 125 << ' ';

cout.unsetf(ios\_base::adjustfield);

cout.precision(2);

cout.setf(ios\_base::uppercase | ios\_base::scientific);

cout << 131.0;

cout.flags(original\_flags);

}

* **123|124..125.1.31E+02**

20. Koji je rezultat izvrsavanja sledeceg programa

#include <iostream>

#include<cmath>

#include<iomanip>

using namespace std;

int main() {

double x, y;

x = 8.0;

y = 3.0;

cout << showpoint << setprecision(2);

cout << x << "^" << y << " = " << pow(x, y) << " " << static\_cast<int>(sqrt(pow(x, y))) << endl;

}

* **8.0^3.0 = 5.1e+02 22**

21.Oznacite linije koda koje su pogresne:

#include <iostream>

using namespace std;

int main() {

int n1{ 300 };

int n2{ 10 };

**catch**

{

**kol = n1 / n2;**

cout << "Kolicnik je " << kol << endl;

}

**try (string exceptionString {**

cout << exceptionString;

}

return 0;

}

**#linije koje su pogresne su oznacene ovako**

22.Sta ce biti prikazano na standardnom izlazu nakon izvrsavanja sledeceg koda  
  
#include <iostream>

using namespace std;

void druga(int x) throw (int)

{

if (x > 1000) throw x;

}

void prva(int x)

{

try

{

druga(1200);

}

catch (...)

{

throw x \* 10;

}

}

int main() {

try

{

prva(10);

}

catch (int vred)

{

cout << vred << endl;

}

return 0;

}

* **100**

23.Oznacite linije koje sadrze greske u sledecem programskom kodu:

void fun(int x) throw ()

{

**if (x < 10) throw 10.0;**

}

int main() {

fun(5);

return 0;

}

24.Operator umetanja u izlazni tok je:

* **<<**

25.Preciznije podesavanje formata etodima klase ios vrsi se sledecim funkcijama:

* **precision()**
* **width()**
* **fill()**

26.Povecanje dimenzija vektora (capacity) cija je postojeca dimenzija N vrsi se:

* **Zavisno od implementacije, za kN elemenata, gde je k obicno 1,5 ili 2**

27. Ponavljanje prijave izuzetka se vrsi naredbom?

* **throw.**

28. Oznacite linije koda koje su pogresne

template <class T1,class T2>

**T1 sum(T1 x, T1 y)**

{

return x + y;

}

int main() {

int n1{ 300 };

double n2{ 30.0 };

**cout << sum(n1, n2) << endl;**

return 0;

}

29. Pametni pokazivaci (smart pointers)

* **Automatski uklanjaju objekte kad na njih vise ne pokazuje nijedan pokazivac**
* **Koriste brojace referenci (reference counting)**

30.Dodavanje novog elementa iza poslednjeg elementa vektora:

* **Zahteva kopiranje postojećih elemenata samo ako je prevaziđen kapacitet vektora (capacity)**
* **Zahteva povećanje dimenzija vektora samo ako je prevaziđen kapacitet vektora (capacity)**

31.osonvne kategorije U/I tokova u jeziku C++ su:

* **Tekstualni**
* **Binarni**

32. Alokatori STL kontejnera:

* **Omogucavaju automatsko se prilagodjavanje STL kontejnera broju elemenata koji se u njih smestaju**
* **Za postojece klase STL kontejnera mogu se definisati sopstveni alokatori**
* **Omogucavaju automatsko se prilagodjavanje STL kontejnera tipu elemenata koji se u njih smestaju**

33.Format podataka u ulazno-izlaznim operacijama moze se precizno definisati

* **Pomocu manipulatorskih funkcija**
* **Pomocu metoda klase ios**

34.Koja od klasa izuzetaka je predefinisana u jeziku C++

* **Overflow\_error**
* **Runtime\_error**
* **Bad\_exeption**
* **Underflow\_error**
* **Logic\_error**
* **Exception**

35.Obelezi ispravne tvrdnje

* Izvrsavanje naredbe “throw” naziva se prijavl;jivanje izuzetka (engl. Throwing an exception)
* Blok catch sadrzi kod koji se izvrsava kada se dogodi izuzetak
* Moze se prijaviti vrednost bilo kojeg tipa

36.Za prijavljivanje izuzetaka u konstruktorima i detruktorima klasa vazi:

* **Konstruktori mogu da prijavljuju izuzetke**
* **Dekonstruktori mogu da prijavljuju izuzetke**

37.U sta spadaju manipulatorske funkcije

* **Mogu se koristiti I kao standardne funkcije, ciji je argument object tipa tok**
* **Menjaju parametre formatiranja tokova**
* **Umecu ili izdvajaju odredjene specijalne znakove**